Recurrence Quantification Analysis

Caterina Ceccato, Jos Prinsen, Ethel Pruss, Anita Vrins

2023-05-27

In the previous entry (Module 6 - Phase Space Reconstruction) a thorough analysis was made, which creates the basis for this entry. Here, we are going to delve into Recurrence Quantification Analysis. This is a method used to analyze the properties of recurrent patterns in time series data, allowing for a comprehensive understanding of the system dynamics and interactions (Webber Jr. & Marwan, 2015). It focuses on the concept of recurrence, which can be described as the degree to which a system comes to inhabit the same or a very similar state. There are several methods that can be used, depending on the amount of time series you want to analyze and on what you want to investigate with the analysis. In this entry we will use 3 different methods: RQA, CRQA and mdCRQA. We are going to start by getting an idea of the dynamics of the singular epochs by applying RQA to all of them. Afterwards, the interaction between a few couples of epochs will be explored. Finally, mdCRQA will be used to delve deeper into the effects of different channels, compared in different epochs.

1) Loading in the data and setting up

The data for 1 (semi) random participant for the adaptive condition is extracted

The data for 1 participant contains 9 lists corresponding to the epochs, and each list contains data for the 8 EEG channels and 3 extra columns that will not be considered in the rest of the analysis.

```
#Check if the dimensions of the data correspond
print(c("Number of epochs: ", length(part_ad)))
```

```
## [1] "Number of epochs: " "9"
```

```
print(c("Number of channels: ", length(part_ad[[1]])))
```

```
## [1] "Number of channels: " "11"
```

```
print("1st epoch")
```

```
## [1] "1st epoch"
```

```
print(part_ad[[1]])
```

```
##             Fz        C3        Cz  ...  AdaptiveRandom  FirstSecond  Participant
## 0   -12.877797 -3.255558 -3.537472  ...             1.0          0.0         39.0
## 1   -10.599298 -2.673650 -2.571720  ...             1.0          0.0         39.0
## 2   -10.229190 -3.916416 -2.177865  ...             1.0         -0.0         39.0
```

```
## 3    -12.108228 -5.393600 -2.425360  ...             1.0          0.0         39.0
## 4    -13.485413 -4.818951 -2.481158  ...             1.0          0.0         39.0
## ...         ...       ...       ...  ...             ...          ...          ...
## 2556  -4.792877 -1.095704  0.331023  ...             1.0          0.0         39.0
## 2557  -3.542809 -1.085884  0.469226  ...             1.0          0.0         39.0
## 2558  -3.196187 -0.469547  0.481530  ...             1.0         -0.0         39.0
## 2559  -4.217521  0.402390  0.354247  ...             1.0          0.0         39.0
## 2560  -5.781861  0.896795  0.301587  ...             1.0          0.0         39.0
##
## [2561 rows x 11 columns]
```

Now that the data is loaded let's start with the first analysis.

2) RQA

RQA stands for Recurrence Quantification Analysis, and it could be seen as the foundation for the methods that will be explored in this entry. In particular, RQA focuses on quantifying uni-variate dynamics, sequences of recurrent patterns, stability, and transitions. This is done by exploring one singular time series.

In order to proceed with applying this method on our data, we first need to fine-tune some settings. In particular, the Tau and Embedding dimension. More information on them and how to perform this step can be found in the entry for module 6. Here they have been calculated directly on the previously loaded dataset.

Another feature that needs to be fine-tuned is the radius in the RQA function. The radius is the threshold we select to decide the maximum distance that there should be between two points to consider them recurrent. We are going to fine tune it by looking at the Recurrence Rate (RR), which is part of the output of RQA. The radius will be adapted so that RR falls between 1 and 5.

After a first analysis, it became evident that it would not be possible to perform RQA on all epochs with the same radius and obtain values of RR between 1 and 5. For this reason, a different radius will be chosen for each epoch. This will mean that the epochs will not be directly comparable, but it is still possible to observe them separately.

```
print(c("Embedding dimension: ", emb.dim))
```

```
## [1] "Embedding dimension: " "9"
```

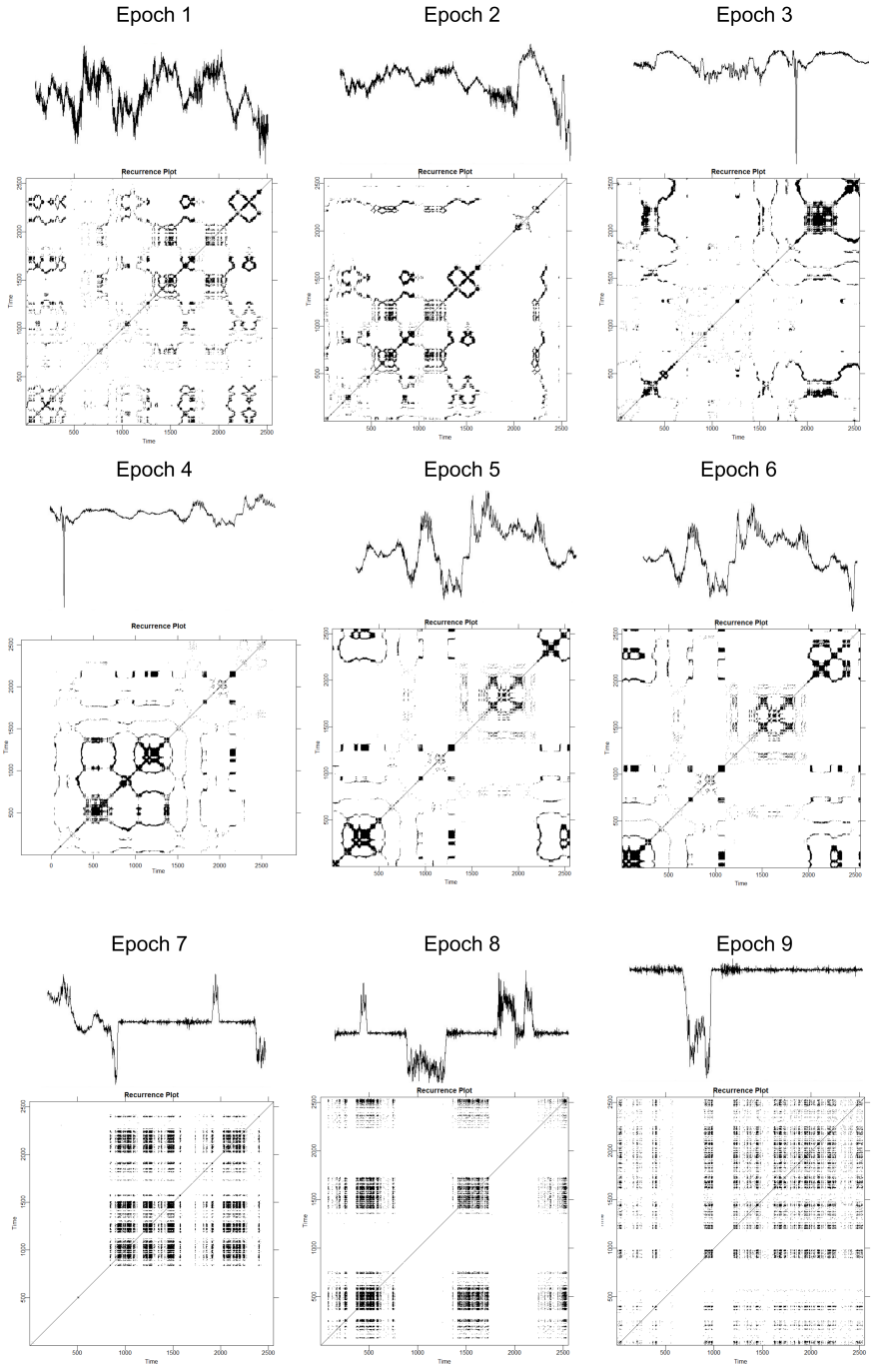
```
participant_rqa <- list()
#Loop over epochs, same procedure as before
for (i in 1:length(part_ad)) {
  epochrqa <- NULL
  chan_avg <- NULL

  for (chan_num in channels) {
    chan <- part_ad[[i]][chan_num]
    chan_avg <- cbind(chan_avg, chan)
  }
  chan_avg <- rowMeans(chan_avg)  # Calculate average across channels

 #set a different radius for different epochs (radii fine-tuned looking at the RR output)
 if (i == 1) {
    radius = 0.92
```

```
  }
  else if (i == 2) {
    radius = 0.51
  }
  else if (i == 3 | i == 5 | i == 6) {
    radius = 0.42
  }
  else if (i == 4) {
    radius = 0.36
  }
  else if ( i == 7 | i == 8) {
    radius = 0.15
  }
  else if ( i == 9) {
    radius = 0.12
  }
  #calculate rqa for each epoch
  epochrqa <- crqa(chan_avg, chan_avg, embed = emb.dim, normalize = 2, rescale = 0,
                   mindiagline = 2, minvertline = 2, radius = radius, tw = .0001,
                   recpt = FALSE, method = "rqa")
  #plot the original timeseries (EEG averaged over channels)
  #plot(chan_avg,type ='l',main= c("Epoch", 1))
  #store the rqas for each epoch in a list
  participant_rqa[[i]] <- epochrqa
}
```

Now that everything is set and rqa has been run, we can explore the recurrence plots for each epoch



As you can see, some

of the plots show different behaviours. In general, the white bands that appear to be in every plot indicate non-stationarity. That is consistent with previous analyses and with the general trend in EEG data. Most of the plots show some pattern that does not seem to be very consistent, and it could be that there is very small correlation if any for those epochs. However, the last three plot show some vertical and horizontal lines, which could be an indication of states changing very slowly, which could be an indication of a certain degree of synchronization. We cannot know for sure whether this difference between plots is an artifact of the different settings used for the radius, but it is still a nice insight in the possible patterns the system might be following.

3) CRQA

Now that the different epochs have been explored separately, it is the more clear that a direct comparison between epochs could be quite insightful. To this end, we will perform Cross Recurrence Quantification analysis. This method builds on RQA and allows for the study of recurrent patterns and dynamic interactions between two different time series (Wallot, S & Leonardi, G., 2018).

Here, we are going to apply CRQA to a few pairs of epochs of the participant selected. This can be considered as a step towards the next analysis, and it is important to notice that only taking extremes of the epochs, such as the first and last epoch, might lead to some information loss in regards to the dynamics over the course of the interaction.

In order to perform CRQA, some parameters need to optimized (the usual tau, embedding dimension and radius). In this case, a useful function exists that helps with that.

```
#select the parameters that will be used to optimize the parameters
par = list(method = "crqa", metric = "euclidean", maxlag =  50,
           radiusspan = 100, radiussample = 40, normalize = 0,
           rescale = 4, mindiagline = 10, minvertline = 10, tw = 0,
           whiteline = FALSE, recpt = FALSE, side = "both",
           datatype = "continuous", fnnpercent  = 20,
           typeami = "mindip", nbins  = 50, criterion = "firstBelow",
           threshold = 1, maxEmb = 20, numSamples = 500,
           Rtol = 10, Atol = 2)
#loop over some of the epochs to get a list containing the average EEG over channels for each epoch
firstandlastepoch <- list()
for (i in c(1,2,5,9)) {
  chan_avg <- NULL  # Initialize chan_avg inside the loop

  for (chan_num in channels) {
    chan <- part_ad[[i]][chan_num]
    chan_avg <- cbind(chan_avg, chan)  # Store channel data in a matrix
  }

  chan_avg <- rowMeans(chan_avg)  # Calculate average across channels
  firstandlastepoch[[i]] <- chan_avg

}
#Calculate the optimal parameters for each pair of epochs

optParams19 <- optimizeParam(firstandlastepoch[[1]], firstandlastepoch[[9]], par,min.rec = 2, max.rec =
print("Parameters for crqa on epoch 1 - epoch 9")
```

```
## [1] "Parameters for crqa on epoch 1 - epoch 9"
```

```
print(unlist(optParams19))
```

```
##     radius     emddim      delay
##  0.2752445  7.0000000 43.0000000
```

```
optParams12 <- optimizeParam(firstandlastepoch[[1]], firstandlastepoch[[2]], par,min.rec = 2, max.rec =
print("Parameters for crqa on epoch 1 - epoch 2")
```

```
## [1] "Parameters for crqa on epoch 1 - epoch 2"
```

```
print(unlist(optParams12))
```

```
##     radius     emddim      delay
##  0.2958088  7.0000000 38.0000000
```

```
optParams15 <- optimizeParam(firstandlastepoch[[1]], firstandlastepoch[[5]], par,min.rec = 2, max.rec =
print("Parameters for crqa on epoch 1 - epoch 5")
```

```
## [1] "Parameters for crqa on epoch 1 - epoch 5"
```

```
print(unlist(optParams15))
```

```
##     radius     emddim      delay
##  0.3026432  7.0000000 49.0000000
```

```
optParams59 <- optimizeParam(firstandlastepoch[[5]], firstandlastepoch[[9]], par,min.rec = 2, max.rec =
print("Parameters for crqa on epoch 5 - epoch 9")
```

```
## [1] "Parameters for crqa on epoch 5 - epoch 9"
```

```
print(unlist(optParams59))
```

```
##     radius     emddim      delay
##  0.1763836  4.0000000 48.0000000
```

Now that the optimal parameters for all the pairs of epochs have been estimated, let's run CRQA.

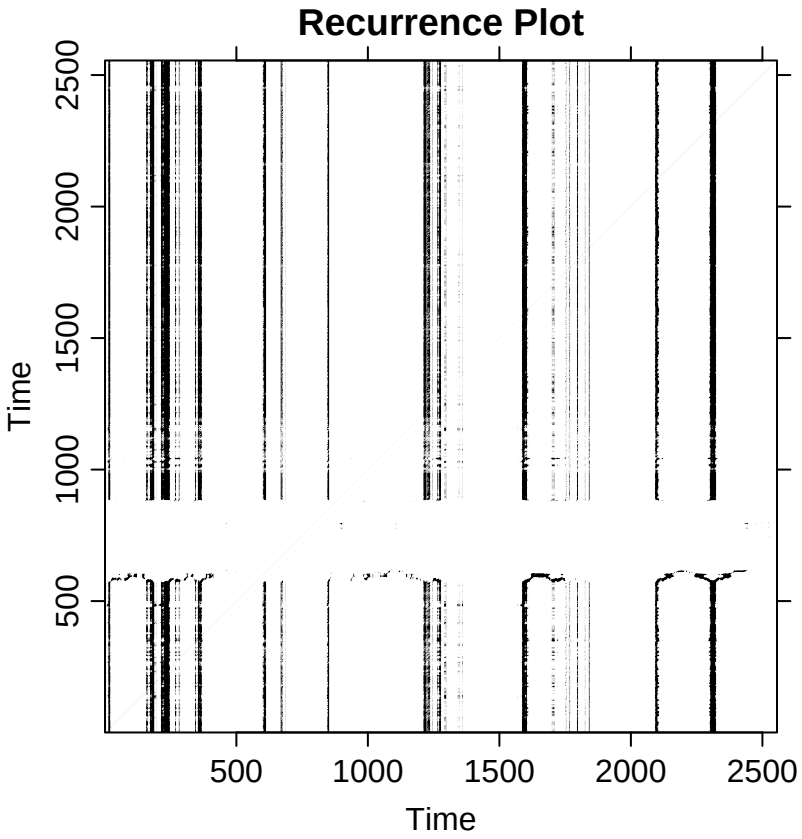
```
# Run CRQA for each pair of epochs and plot the recurrence plot
crqa19 <- crqa(firstandlastepoch[[1]], firstandlastepoch[[9]], embed = optParams19$emddim, normalize = 1

crqa19[1:5]
```

```
## $RR
## [1] 8.640714
##
## $DET
## [1] 95.74785
```

```
##
## $NRLINE
## [1] 63885
##
## $maxL
## [1] 57
##
## $L
## [1] 8.453988
```

```
RP19 <- as.matrix(crqa19$RP)
recurrence_plot(RP19)
```



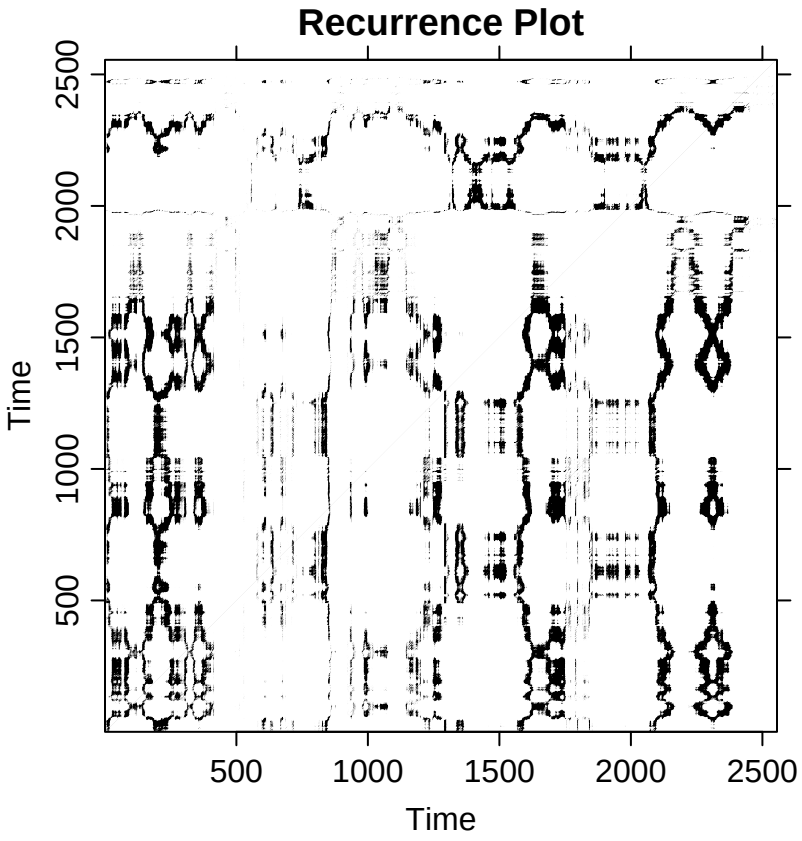

```
crqa12 <- crqa(firstandlastepoch[[1]], firstandlastepoch[[2]], embed = optParams12$emddim, normalize = 1

crqa12[1:5]
```

```
## $RR
## [1] 10.97514
##
## $DET
## [1] 97.08232
##
## $NRLINE
## [1] 63623
```

```
##
## $maxL
## [1] 163
##
## $L
## [1] 10.93246
```

```
RP12 <- as.matrix(crqa12$RP)
recurrence_plot(RP12)
```



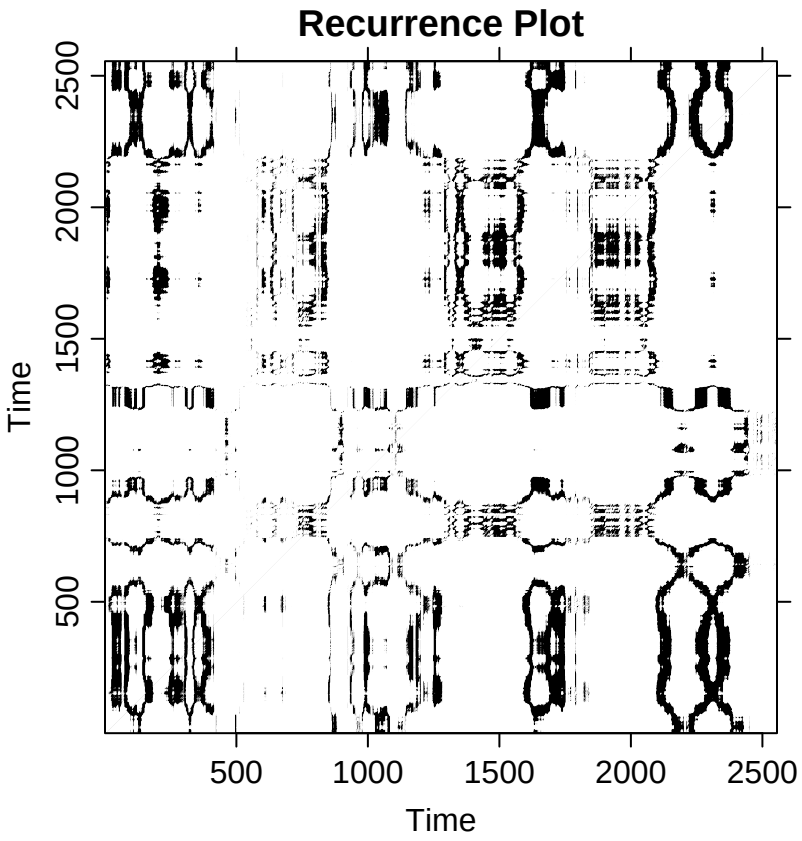

```
crqa15 <- crqa(firstandlastepoch[[1]], firstandlastepoch[[5]], embed = optParams15$emddim, normalize = 1

crqa15[1:5]
```

```
## $RR
## [1] 12.69888
##
## $DET
## [1] 97.8458
##
## $NRLINE
## [1] 64779
##
## $maxL
## [1] 159
```

```
##
## $L
## [1] 12.52147
```

```
RP15 <- as.matrix(crqa15$RP)
recurrence_plot(RP15)
```

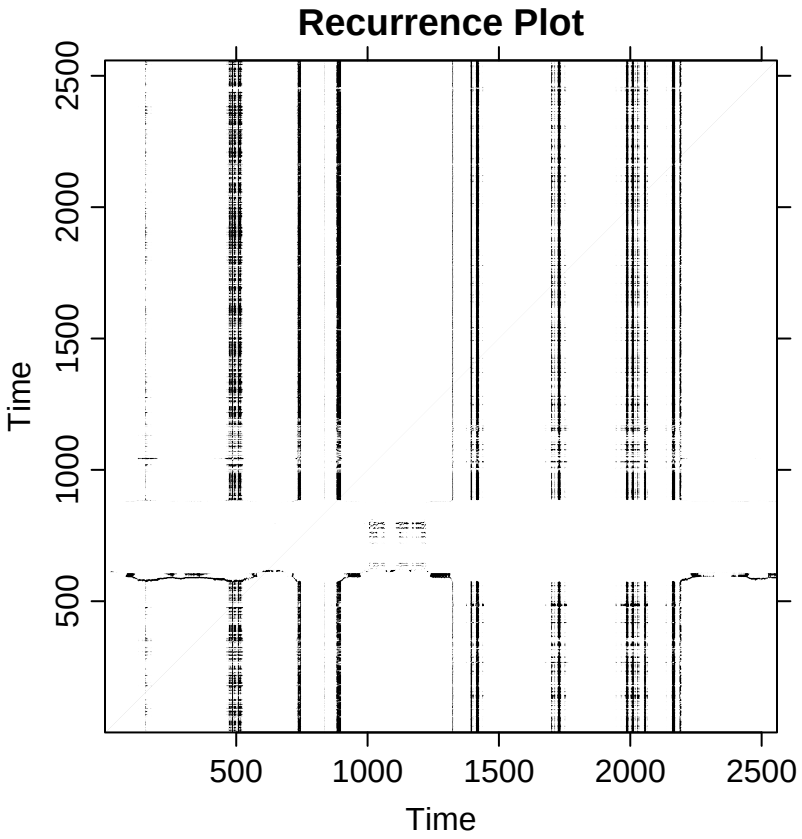



```
crqa59 <- crqa(firstandlastepoch[[5]], firstandlastepoch[[9]], embed = optParams59$emddim, normalize = 1

crqa59[1:5]
```

```
## $RR
## [1] 5.968383
##
## $DET
## [1] 96.34653
##
## $NRLINE
## [1] 49962
##
## $maxL
## [1] 36
##
## $L
## [1] 7.531024
```

```
RP59 <- as.matrix(crqa59$RP)
recurrence_plot(RP59)
```



As it can be seen in the plots above, we decided to apply crqa to 4 pairs: 1) Epoch 1 - Epoch 9 2) Epoch 1 - Epoch 2 3) Epoch 1 - Epoch 5 4) Epoch 5 - Epoch 9

For the first pair, the vertical lines could indicate a slow change over time, which would make sense, considering that this pair compares the first and last epoch. As for the second and third plots, there seems to be some pattern that might indicate cyclicities. This could mean that there is some rhytmic activity in the system or that there are periodic phenomena. Considering that the data was taken from a learning task, it could be that these patterns show some degree of learning. However, the last pair that we investigated, pertaining the 5th and last epoch, shows a pattern that is more similar to the first one. Maybe this could indicate a change in the dynamics, or it could be a sign that only considering the last part of the task leads to the loss of important information about the learning process.

4) mdCRQA

Now that we have explored the overall dynamics comparing different couples of epochs, we can proceed to a more in depth analysis of the effect of the different channels in certain epochs. In order to do this, we will use mdCRQA. This method extends CRQA to analyze the cross-recurrence of multiple variables simultaneously, providing insights into the relationships, patterns, and dynamics among different dimensions in a dataset (Wallot, S., 2019). Given that a lot of information is certainly lost when averaging over channels (what was done so far in this entry), mdCRQA can be a useful tool to get more information on what explored so far. As usual, we first need to set the right parameters, and we will then proceed with the analysis.

```
#loop over epochs to get a list of all the EEG channels for each epoch
epochs_chans <- list()
for (i in c(1,2,5,9)) {
  chan_avg <- NULL  # Initialize chan_avg inside the loop

  for (chan_num in channels) {
    chan <- part_ad[[i]][chan_num]
    chan_avg[[chan_num]] <- chan  # Store channel data in a matrix
  }

  #chan_avg <- rowMeans(chan_avg)  # Calculate average across channels
  epochs_chans[[i]] <- chan_avg

}
#transform the data in the right format
epoch1_chans <- do.call(cbind, epochs_chans[[1]])
epoch2_chans <- do.call(cbind, epochs_chans[[2]])
epoch5_chans <- do.call(cbind, epochs_chans[[5]])
epoch9_chans <- do.call(cbind, epochs_chans[[9]])




# Set parameters for optimize function
par = list(method = "mdcrqa", metric = "euclidean", maxlag =  100,
           radiusspan = 100, radiussample = 40, normalize = 0,
           rescale = 4, mindiagline = 10, minvertline = 10, tw = 0,
           whiteline = FALSE, recpt = FALSE, side = "both",
           datatype = "continuous", fnnpercent  = NA,
           typeami = NA, nbins  = 200, criterion = "firstBelow",
           threshold = 5, maxEmb = 20, numSamples = 500,
           Rtol = 10, Atol = 2)

#run the optimize function for each pair of epochs

optParams19_chans <- optimizeParam(epoch1_chans, epoch9_chans, par,min.rec = 2, max.rec = 5)
print("Parameters for mdcrqa on epoch1 - epoch9")
```

```
## [1] "Parameters for mdcrqa on epoch1 - epoch9"
```

```
print(unlist(optParams19_chans))
```

```
##    radius    emddim     delay
## 0.4553202 2.0000000 1.0000000
```

```
optParams12_chans <- optimizeParam(epoch1_chans, epoch2_chans, par,min.rec = 2, max.rec = 5)
print("Parameters for mdcrqa on epoch1 - epoch2")
```

```
## [1] "Parameters for mdcrqa on epoch1 - epoch2"
```

```
print(unlist(optParams12_chans))
```

```
##    radius    emddim     delay
## 0.3895833 2.0000000 1.0000000
```

```
optParams15_chans <- optimizeParam(epoch1_chans, epoch5_chans, par,min.rec = 2, max.rec = 5)
print("Parameters for mdcrqa on epoch1 - epoch5")
```

```
## [1] "Parameters for mdcrqa on epoch1 - epoch5"
```

```
print(unlist(optParams15_chans))
```

```
##    radius    emddim     delay
## 0.4332001 2.0000000 1.0000000
```

```
optParams59_chans <- optimizeParam(epoch5_chans, epoch9_chans, par,min.rec = 2, max.rec = 5)
print("Parameters for mdcrqa on epoch5 - epoch9")
```

```
## [1] "Parameters for mdcrqa on epoch5 - epoch9"
```

```
print(unlist(optParams59_chans))
```

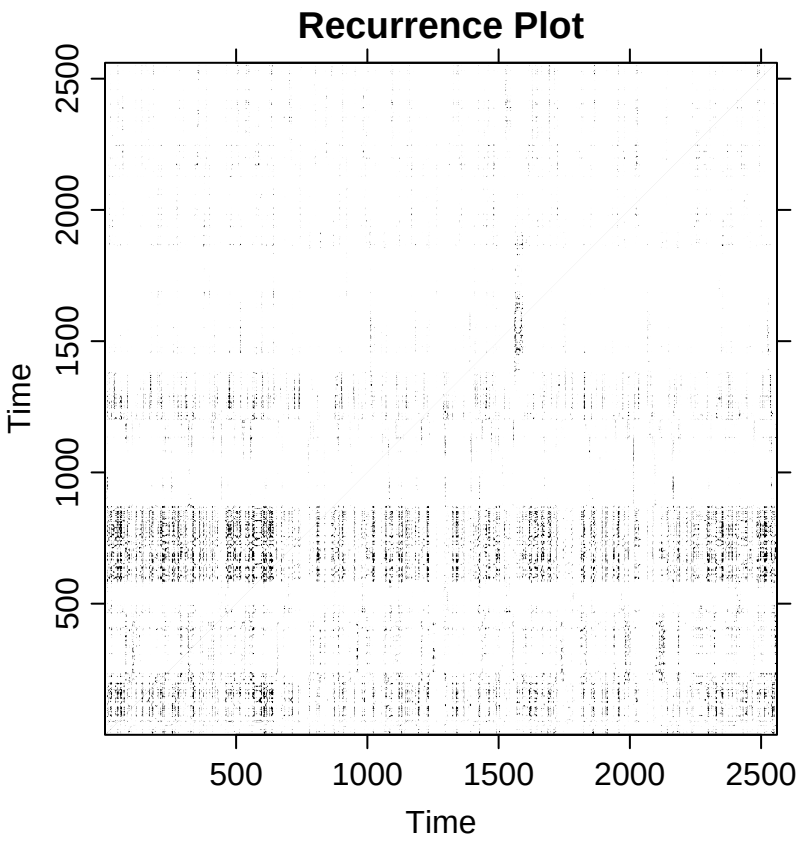
```
##    radius    emddim     delay
## 0.4568631 2.0000000 1.0000000
```

```
#Run mdcrqa for each pair of epochs and plot the recurrence plot
mdcrqa19 <- crqa(epoch1_chans, epoch9_chans, embed = optParams19_chans$emddim, normalize = 2, rescale =

mdcrqa19[1:5]
```

```
## $RR
## [1] 3.559982
##
## $DET
## [1] 86.36046
##
## $NRLINE
## [1] 61686
##
## $maxL
## [1] 23
##
## $L
## [1] 3.2663
```

```
RP19_chans <- as.matrix(mdcrqa19$RP)
recurrence_plot(RP19_chans)
```

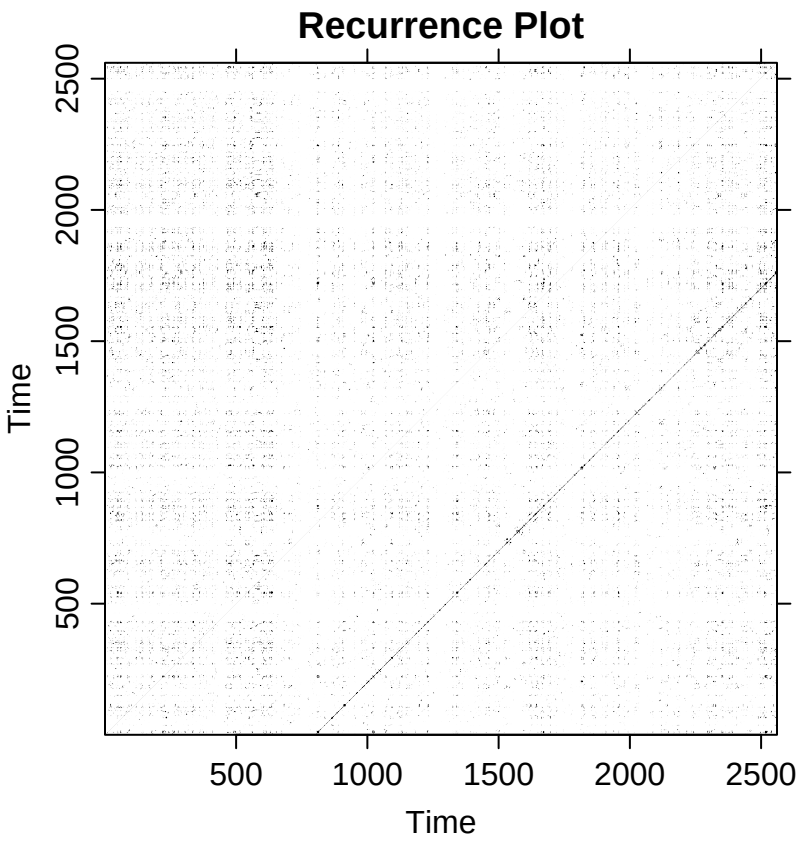



```
mdcrqa12 <- crqa(epoch1_chans, epoch2_chans, embed = optParams12_chans$emddim, normalize = 2, rescale =

mdcrqa12[1:5]
```

```
## $RR
## [1] 3.162766
##
## $DET
## [1] 74.82041
##
## $NRLINE
## [1] 55762
##
## $maxL
## [1] 1760
##
## $L
## [1] 2.781177
```

```
RP12_chans <- as.matrix(mdcrqa12$RP)
recurrence_plot(RP12_chans)
```

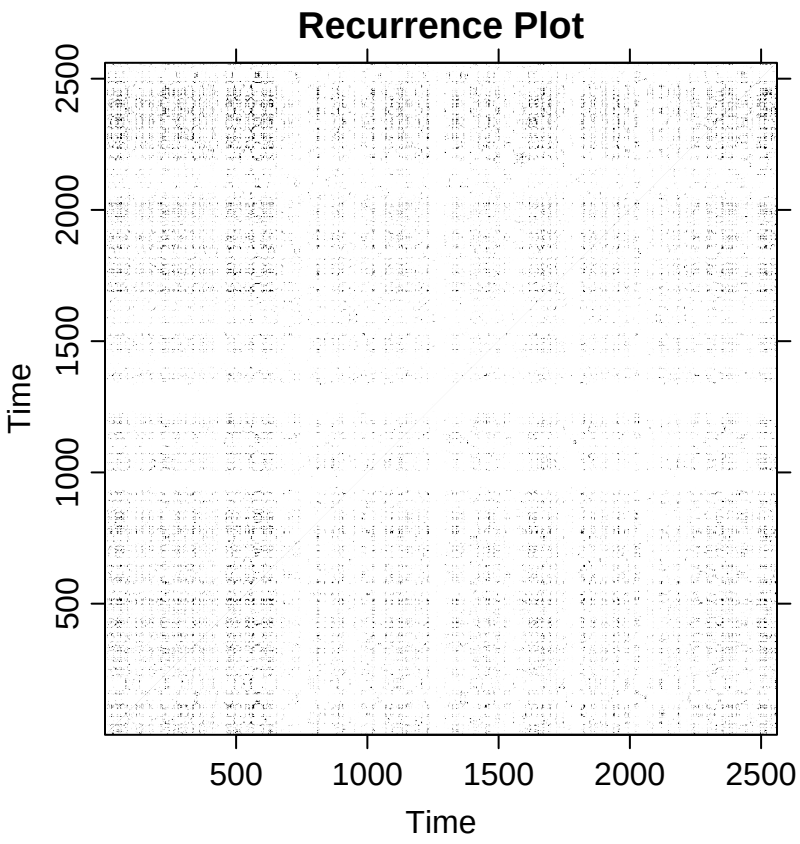



```
mdcrqa15 <- crqa(epoch1_chans, epoch5_chans, embed = optParams15_chans$emddim, normalize = 2, rescale =

mdcrqa15[1:5]
```

```
## $RR
## [1] 4.678207
##
## $DET
## [1] 79.62823
##
## $NRLINE
## [1] 85028
##
## $maxL
## [1] 23
##
## $L
## [1] 2.871207
```

```
RP15_chans <- as.matrix(mdcrqa15$RP)
recurrence_plot(RP15_chans)
```

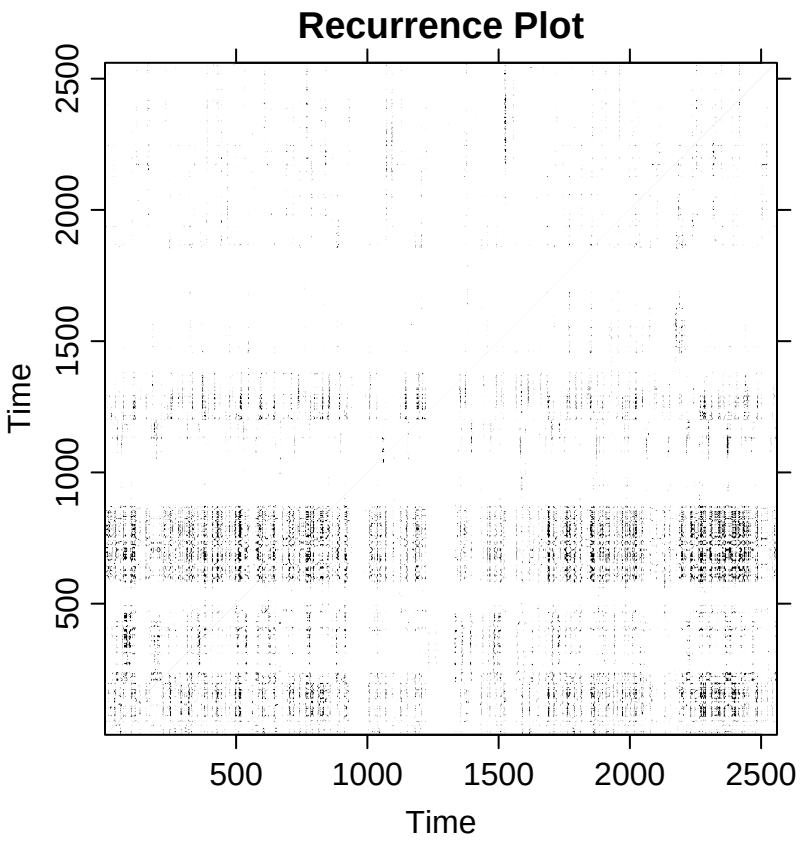



```
mdcrqa59 <- crqa(epoch5_chans, epoch9_chans, embed = optParams59_chans$emddim, normalize = 2, rescale =

mdcrqa59[1:5]
```

```
## $RR
## [1] 3.807327
##
## $DET
## [1] 88.32104
##
## $NRLINE
## [1] 64800
##
## $maxL
## [1] 25
##
## $L
## [1] 3.400864
```

```
RP59_chans <- as.matrix(mdcrqa59$RP)
recurrence_plot(RP59_chans)
```



These plots show results that are comparable to what seen for crqa. That is understandable, as the only difference between the two analyses is that for crqa, since it is not possible to compare multidimensional systems, the average of the channels was taken, whereas here all the channels are considered. In the first plot, some vertical lines can be seen at the bottom, but none at the top. This could mean that there are some recurrent patterns in the first epoch that do not repeat in the last one. This could be supported by the relatively high value of L (3.2663), indicating relatively high complexity. This would mean that the EEG channels exhibit a rich and diverse set of patterns and structures, which could possibly be further analyzed. As for the second plot, the presence of random dots all over the plot could indicate noise and lack of recurrent patterns. However, the single diagonal line on the bottom right could provide some interesting information. That could indicate a temporary or transient phenomenon occurring between the first and second epochs. There might be a specific event or state that affects the dynamics of the system during this period, leading to a unique recurrent pattern. The third plot could indicate high variability, due to the lack of patterns. However, the high determinism score (79) and the high number of recurrence lines (85028) might indicate a degree of repeatability and instances of recurrent states or behaviors between the two epochs. It seems like the system's behavior has a mix of both recurrent and non-recurrrent states, therefore leading to a disordered recurrence plot. As for the last plot, the dynamics seem to be similar to what could be seen in the first plot. It is worth noting that both the first and last plots compare two epochs that are relatively distant to each other (more so for the first one). The presence of similar patterns for these two plots might be an indication of an overall dynamic that does not show clearly the more complex dynamics that the system goes through over time.

5) Conclusion

This module has explored recurrence quantification analysis by considering 3 methods: rqa, crqa and mdcrqa. First, an analysis of the recurrence in the singular epochs was done, which showed that over time (epochs)

the system seemed to go from a more cyclic behaviour to a more stable one, which could be correlated to the learning process of the participant. As second analysis, we compared 4 pairs of epochs to see investigate their recurrence. We found out that when considering epochs closer together some more cyclical, complex patterns seem to emerge, whereas in epochs that are more distant the dynamics are more stable. This was also validated in the last analysis that was performed. In the first two analysis the average over channels was used, whereas in the last one all the channels were considered. mdCRQA was applied to the same pairs of epochs observed earlier and, as already mentioned, the overall analysis seemed to validate what previosuly seen with crqa. It is important to notice that all channels have been considered, but in a future analysis, it could be interesting to only consider some specific channels. In conclusion, this entry showed that recurrence quantification analysis can be a useful tool to explore the dynamics of EEG in a learning task. Future analyses could include different data, more targeted to a specific task, or the analysis of different components of this data, possibly including comparisons between participants and conditions.

References

Coco, M. I., and Dale, R. (2014). Cross-recurrence quantification analysis of categorical and continuous time series: an R package. Frontiers in psychology, 5, 510.

H. Kantz and T. Schreiber: Nonlinear Time series Analysis (Cambridge university press)

Marwan, N., Romano, M. C. Theil, M., & Kurths, J. (2007). Recurrence plots for the analysis of complex systems. Physics Reports, 438, 237-329.

Wallot, S., Roepstorff, A., & Mønster, D. (2016). Multidimensional Recurrence Quantification Analysis (MdRQA) for the analysis of multidimensional time-series: A software implementation in MATLAB and its application to group-level data in joint action. Frontiers in psychology, 7, 1835. Webber Jr., C. L., & Marwan, N. (Eds.). (2015). Recurrence quantification analysis: Theory and best practices. Springer

Wallot, S., & Leonardi, G. (2018). Analyzing multivariate dynamics using cross-recurrence quantification analysis (crqa), diagonal-cross-recurrence profiles (dcrp), and multidimensional recurrence quantification analysis (mdrqa)–a tutorial in r. Frontiers in Psychology, 9, 2232.

Wallot, S. (2019). Multidimensional Cross-Recurrence Quantification Analysis (MdCRQA)–a method for quantifying correlation between multivariate time-series. Multivariate behavioral research, 54(2), 173-191

Webber, C. L., & Zbilut, J. P. (2005). Recurrence quantification analysis of nonlinear dynamical time series. In S. Riley and G. C. Van Orden (eds). Tutorials in contemporary nonlinear methods for the behavioral sciences.